UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/785,564	02/24/2004	Britton Worth Piehler	BEAS-1439US1	1341

23910 7590 04/02/2008

FLIESLER MEYER LLP
650 CALIFORNIA STREET
14TH FLOOR
SAN FRANCISCO, CA 94108

EXAMINER	
YIGDALL, MICHAEL J	
ART UNIT	**PAPER NUMBER**
2192	

MAIL DATE	DELIVERY MODE
04/02/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/785,564	PIEHLER ET AL.	
	Examiner	Art Unit	
	Michael J. Yigdall	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *13 December 2007*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1,6-12,14-26 and 28-30* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1,6-12,14-26 and 28-30* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *12/13/2007*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on December 13, 2007 has been entered. Claims 1, 6-12, 14-26 and 28-30 are pending.

Response to Amendment

2. The objection to claim 9 has been withdrawn in view of Applicant's amendment.

3. The rejection of claims 1 and 6-31 under 35 U.S.C. 101 has been withdrawn in view of Applicant's amendment.

Response to Arguments

4. Applicant's arguments have been fully considered but they are not persuasive.

In response to Applicant's arguments, the examiner notes that the test for obviousness is not that the claimed invention must be expressly suggested in any one or all of the references. Rather, the test is what the combined teachings of the references would have suggested to those of ordinary skill in the art. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981).

Applicant contends, generally, "Skinner does not disclose maintaining a type cache containing signatures of classes" (remarks, page 9).

However, Chan teaches accessing Java classes (see, for example, paragraph [0059]). Skinner teaches that such Java classes are loaded on demand from the network or from a local file system (see, for example, column 7, lines 40-47). Furthermore, as set forth in the Office action, Skinner teaches maintaining an object cache containing the states of Java data objects (see, for example, column 21, lines 5-17). One of ordinary skill in the art could, with predictable results, incorporate such a cache into Chan. Indeed, one of ordinary skill in the art would have been motivated to implement Chan such that the Java classes are accessed from a type cache rather than loaded from the network or elsewhere. Accordingly, the combined teachings of the references would have suggested the claimed subject matter to those of ordinary skill in the art.

Applicant contends, generally, "Iwashita does not disclose a thread pool that allows compilation of multiple files to be performed in parallel" (remarks, pages 9-10).

Nonetheless, Applicant acknowledges that Iwashita teaches the compilation of multiple files (remarks, page 9). Indeed, Iwashita teaches compiling multiple files simultaneously, which is to say, in parallel (see, for example, column 15, lines 34-35). One of ordinary skill in the art could, with predictable results, implement Abrams so as to compile multiple files in parallel, as Iwashita suggests. Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate into Abrams a thread pool that allows such compilation to be performed. Accordingly, the combined teachings of the references would have suggested the claimed subject matter to those of ordinary skill in the art.

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claims 1, 6-8, 11, 12, 14-26 and 28-30 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent No. 6,836,883 to Abrams et al. (art of record, "Abrams") in view of U.S. Pub. No. 2003/0028364 to Chan et al. (art of record, "Chan") in view of U.S. Pub. No. 2004/0103406 to Patel (art of record, "Patel") in view of U.S. Patent No. 7,073,167 to Iwashita (art of record, "Iwashita") and in view of U.S. Patent No. 6,721,740 to Skinner et al. (art of record, "Skinner").

With respect to claim 1 (currently amended), Abrams teaches a system comprising:

one or more compilers, executed by one or more processors (see at least col.2:1-6; different front ends, different source languages, front end compiler system col.2:20-60; 56, 58, 60 FIG.3 & associated text), wherein the compilers support mixing and nesting of languages within source file (see at least metadata, instruction code col.2:20-60; 84, 90, 92 FIG.4 & associated text);

an extensible multi-language compiler framework (see at least multiple front ends, multiple programming languages col.2:20-60), wherein the compiler framework provides a source code editor with information about the source file comprising: signature of classes

defined by the source file and errors found in the source file and information exposed by any languages (see at least compilers, front end analysis, parses, source language file, lexical, grammatical, syntactic problems, reporting errors, libraries, "include", "import" statements col.1:13-67), wherein language-independent source code editor communicates to the compiler framework using language-independent metadata (see at least front end 22, common language file 26, metadata 28 col.4:34-col.5:29; 28, 33 FIG.1 & associated text).

Abrams does not expressly disclose said source code editor as a language-independent source code editor. However, Chan teaches a language-independent source code editor that supports the mixing and nesting of languages within a source file (see at least 15, 16 FIG.1 & associated text) with information about the source file, comprising: signatures of classes defined by the source file (see at least 20 FIG.2 & associated text), errors found in the source file (see at least 49, 54-55 FIG.4 & associated text), stack of nested languages at any point in the source file (see at least 15, 16 FIG.1 & associated text), and information exposed by any languages (see at least 14-17 FIG.1 & associated text).

Abrams and Chan are analogous art because they are both directed to software development. It would have been obvious to one of ordinary skill in the pertinent art at the time the invention was made to incorporate the teaching of Chan into that of Abrams for the inclusion of the language-independent source code editor. And the motivation for doing so would have been to provide development assistance for all of the languages used in a hybrid or mixed language source file developed in an IDE (see at least Chan paragraph [0010]).

Abrams does not expressly disclose wherein the compiler framework has error correction in code-generation, permitting a user to run code even if there is an error in the code. However,

Patel teaches wherein the compiler framework has error correction in code-generation, permitting users to run their code even if there is an error in it (see at least paragraphs [0006], [0027]-[0028] and [0032]). It would have been obvious to one of ordinary skill in the pertinent art at the time the invention was made to incorporate the teaching of Patel into that of Abrams for the inclusion of error correction in code-generation. And the motivation for doing so would have been to alleviate the need for the programmer to identify missing and/or corrupt files during compilation (see at least Patel paragraph [0005]).

Abrams does not expressly disclose wherein a thread pool allows compilation of multiple files to be performed in parallel. However, Iwashita teaches a system and method for parallel compilation of multiple files (see at least FIG.24 & associated text; FIG.22 & associated text; col.15:34-36). It would have been obvious to one of ordinary skill in the pertinent art at the time the invention was made to incorporate the teaching of Iwashita into that of Abrams for the inclusion of parallel compilation. And the motivation for doing so would have been to reduce duplicate (intermediate or executable) code, thereby reducing the total amount of code generated by the compiler (see at least Iwashita col.14:65-col.15:3).

Chan does not expressly disclose wherein a type cache contains signatures for classes. However, Skinner teaches a system and method for performing active update of Java objects (see at least Abstract; Java classes, JavaBeans col.7:15-67) wherein the states of the Java objects (i.e., Java signatures of classes) are serialized and stored in the type cache (see at least 305B, serializes data objects, object cache 303A col.21:5-17; 303A, 303B FIG.3 & associated text). It would have been obvious to one of ordinary skill in the pertinent art at the time the invention was made to incorporate the teaching of Skinner into that of Chan for the inclusion of a type cache.

And the motivation for doing so would have been maintain (i.e., serialize and save in the type cache) a list of Java objects that are undergoing modification (i.e., being edited in Chan's environment) in order to provide efficient and real-time update notifications multiple programmers/developers/users across a multi-tier network (see at least Skinner col.1:45-65; col.8:13-51).

With respect to claim 6 (previously presented), the rejection of base claim 1 is incorporated. Chan further teaches wherein integrating a new language does not require separate instructions to enable compiling or editing of the new language (see at least paragraph [0034]).

With respect to claim 7 (previously presented), the rejection of base claim 1 is incorporated. Chan further teaches wherein the language-independent source code editor displays errors for mismatched start and end XML tags embedded in the source code and performs auto-completion of XML tags embedded in the source code (see at least FIG.3 & associated text; 17 FIG.1 & associated text).

With respect to claim 8 (previously presented), the rejection of base claim 7 is incorporated. Chan further teaches wherein the language-independent source code editor displays errors and performs auto-completion independent of the host language embedding XML tags (see at least 17 FIG.1 & associated text; paragraph [0056]; 49, 53-54 FIG.4 & associated text).

With respect to claim 11 (previously presented), the rejection of base claim 1 is incorporated. Chan further teaches wherein the compiler framework enables the language-

independent source code editor to provide visual indication of errors throughout a source file with mixed languages (see at least FIG.3 & associated text; 49, 53-54 FIG.4 & associated text)

With respect to claim 12 (previously presented), the rejection of base claim 1 is incorporated. Chan further teaches wherein the compiler framework keeps track of errors in source files in a project so that a user can have a list of errors in opened and unopened source code files in a project (see at least 55 FIG.4 & associated text; paragraphs [0050], [0054]).

With respect to claim 14 (previously presented), the rejection of base claim 1 is incorporated. Chan further teaches wherein the compiler framework allows an outer language compiler to pass of processing of a section of a document to an inner language compiler (see at least 15-16 FIG.1 & associated text; 49, 53 FIG.4 & associated text).

With respect to claim 15 (previously presented), the rejection of base claim 14 is incorporated. Chan further teaches wherein a parse tree produced by the inner compiler is available to the outer compiler (see at least paragraphs [0003], [0046], [0047] and [0050]).

With respect to claim 16 (previously presented), the rejection of base claim 15 is incorporated. Chan further teaches wherein either the inner compiler or the outer compiler can determine where the span of the inner compiler's language content ends (see at least 15-16 FIG.1 & associated text; 49, 53 FIG.4 & associated text).

With respect to claim 17 (previously presented), the rejection of base claim 1 is incorporated. Chan further teaches wherein the compiler framework includes a parser generator

and a scanner generator (see at least scanner programs S1, S2, S3 paragraph [0013]; parser programs P paragraph [0014]).

With respect to claim 18 (previously presented), the rejection of base claim 17 is incorporated. Chan further teaches wherein generated parsers are able to recover from all single token errors and missing identifiers that occur during code completion (see at least paragraphs [0003], [0036]).

With respect to claim 19 (previously presented), the rejection of base claim 1 is incorporated. Chan further teaches wherein the compiler framework provides the source code editor with names of classes and packages in a project and errors found in any source files in a project (see at least FIG.1 & associated text).

With respect to claim 20 (previously presented), the rejection of base claim 19 is incorporated. Chan further teaches wherein the compiler framework is notified of a change in a file, the information about the file is updated within a time limit for a single-file recompile (see at least paragraph [0043]).

With respect to claim 21 (previously presented), the rejection of base claim 20 is incorporated. Chan further teaches wherein after the file is recompiled, the compiler framework provides the source code editor with a list of changes that occurred to the file information (see at least paragraph [0058]).

With respect to claim 22 (currently amended), the rejection of base claim 1 is incorporated. Chan further teaches wherein the compiler framework includes a project compiler,

wherein the project compiler contains a list of source directories and the class path (see at least paragraphs [0003], [0006]-[0008], [0059]). Skinner further teaches maintaining a type cache which contains signatures of classes in the project (see at least 305B, serializes data objects, object cache 303A col.21:5-17; 303A, 303B FIG.3 & associated text).

With respect to claim 23 (previously presented), the rejection of base claim 22 is incorporated. Skinner further teaches wherein the type cache is indexed by file name and by class name, and maintains a current list of errors and a list of dependencies (see at least 304A FIG.3 & associated text; col.16:5-20).

With respect to claim 24 (previously presented), the rejection of base claim 23 is incorporated. Skinner further teaches wherein the project compiler and the type cache are serializable (see at least 305B, serializes data objects, object cache 303A col.21:5-17; 303A, 303B FIG.3 & associated text).

With respect to claim 25 (previously presented), the rejection of base claim 1 is incorporated. Chan further teaches wherein a file compiler is used to perform compilation of a single source file (see at least paragraphs M file [0039], [0041]).

With respect to claim 26 (previously presented), the rejection of base claim 25 is incorporated. Abrams further teaches wherein the file compiler supports interoperation of different languages by using a common intermediate language (see at least col.4:34-col.5:1).

With respect to claim 28 (previously presented), the rejection of base claim 25 is incorporated. Chan further teaches wherein the file compiler remembers where the language nesting occurs for reuse on subsequent parses (see at least 15-16 FIG.1 & associated text).

With respect to claim 29 (previously presented), the rejection of base claim 28 is incorporated. Chan further teaches wherein the outer language implements a name resolution interface to allow the inner language to resolve references to names defined outside of the nest language (see at least paragraph [0056]).

With respect to claim 30 (previously presented), the rejection of base claim 1 is incorporated. Chan further teaches wherein all parsing is performed on background threads (see at least scanner programs S1, S2, S3 paragraph [0013]; parser programs P paragraph [0014]; paragraph [0056]).

7. Claim 9 is rejected under 35 U.S.C. 103(a) as being unpatentable over Abrams in view of Chan in view of Patel in view of Iwashita and in view of Skinner, as applied to claim 1 above, and further in view of U.S. Pub. No. 2003/0023957 to Bau, III et al. (art of record, "Bau").

With respect to claim 9 (currently amended), the rejection of base claim 1 is incorporated. Abrams does not expressly disclose wherein the language-independent source code editor provides syntax coloring and code completion for editing annotations. However, Bau teaches a compiler for parsing (i.e., editing) source file comprising annotations (see at least paragraphs [0023]; [0036]-[0037], [0048] and [0070]). It would have been obvious to one of ordinary skill in the pertinent art at the time the invention was made to incorporate the teaching of Bau into that

of Chan & Abrams for the inclusion of annotations. And the motivation for doing so would have been to simply the developer's task and enable client software written in variety of languages to interact with web services and for the services to interact with other external services (see at least Bau paragraphs [0005], [0007] and [0023]).

8. Claim 10 is rejected under 35 U.S.C. 103(a) as being unpatentable over Abrams in view of Chan in view of Patel in view of Iwashita and in view of Skinner, as applied to claim 1 above, and further in view of U.S. Patent No. 6,367,068 to Vaidyanathan et al. (art of record, "Vaidyanathan").

With respect to claim 10 (previously presented), the rejection of base claim 1 is incorporated. Abrams does not expressly disclose wherein the compiler framework makes it possible to reparse in near real-time with no performance degradation noticeable to the user. However, Vaidyanathan teaches a source code editor (see at least 202, 206 FIG.2 & associated text) and a dynamic parser (see at least 204, 200 FIG.2 & associated text) for parsing the source code as the source code is being edited (see at least) wherein the dynamic parser reparses the source code in near real-time with no performance degradation noticeable to the user (see at least real-time parsing col.1:12-15; col.2:15-30; static parsers, dynamic parsers, indeterminate times col.7:38-65). It would have been obvious to one of ordinary skill in the pertinent art at the time the invention was made to incorporate the teaching of Vaidyanathan into that of Abrams for the inclusion of near real-time reparsing. And the motivation for doing so would have been to eliminate the need for explicit compiling (i.e., reparsing) by the programmer in order to identify

errors in the source code which is being edited, as well as to facilitate code auto-completion (see at least Vaidyanathan col.1:42-55; col.2:37-47).

Conclusion

9. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Michael J. Yigdall whose telephone number is 571-272-3707. The examiner can normally be reached on Monday to Friday from 8:00 AM to 4:30 PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on 571-272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Michael J. Yigdall
Examiner
Art Unit 2192

/Michael J. Yigdall/